

## PATENT ABSTRACTS OF JAPAN

(11)Publication number : 2002-268896

(43)Date of publication of application : 20.09.2002

(51)Int.Cl.

G06F 9/45

G05B 19/05

(21)Application number : 2001-068674

(71)Applicant : HITACHI LTD

(22)Date of filing : 12.03.2001

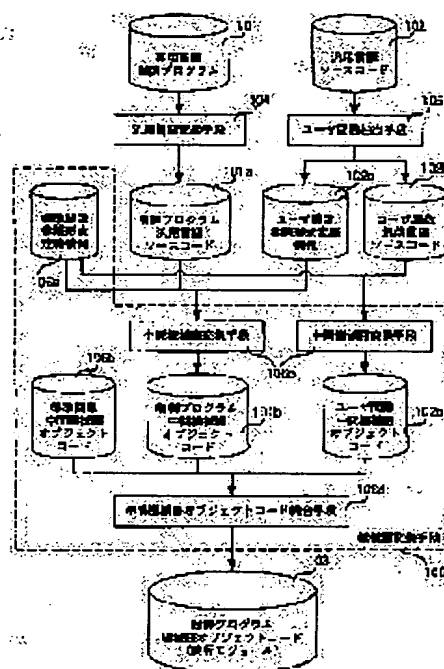
(72)Inventor : KOYAMA MASAHIRO  
HOSODA YUJI  
MATSUSHITA TSURUMASA  
SHIMOZU TADAO

## (54) METHOD AND DEVICE FOR GENERATING CONTROL PROGRAM

## (57)Abstract:

**PROBLEM TO BE SOLVED:** To provide a method for generating a control program by which wasteful labor and time for quoting a user function is relieved and an input mistake is reduced.

**SOLUTION:** A user function extracting means 105 generates the reference form definition information 102a of the user function to be utilized in the control program and a user function general purpose language source code 102b from a source code 102 which is described by a user through the use of a general purpose language. A general purpose language converting means 104 converts the control program 101 where the description of the user function is mixed into a control program general purpose language source code 101a and a machine language converting means 106 converts the source code 101a into a machine language through the use of definition information 102a and the source code 102b.



## LEGAL STATUS

[Date of request for examination]

[Date of sending the examiner's decision of rejection]

[Kind of final disposal of application other than the examiner's decision of rejection or application converted registration]

[Date of final disposal for application]

[Patent number]

[Date of registration]

[Number of appeal against examiner's decision of rejection]

[Date of requesting appeal against examiner's decision of rejection]

[Date of extinction of right]

Copyright (C); 1998,2003 Japan Patent Office

(19) 日本国特許庁 (J P)

(12) 公開特許公報 (A)

(11) 特許出願公開番号  
特開2002-268896  
(P2002-268896A)

(43) 公開日 平成14年9月20日 (2002.9.20)

(51) Int.Cl. <sup>7</sup>	識別記号	F I	テレポート (参考)
G 0 6 F 9/45		G 0 6 F 9/44	3 2 2 M 5 B 0 8 1
G 0 5 B 19/05		G 0 5 B 19/05	A 5 H 2 2 0

審査請求 未請求 請求項の数 5 O L (全 13 頁)

(21) 出願番号 特願2001-68674 (P2001-68674)  
(22) 出願日 平成13年3月12日 (2001.3.12)

(71) 出願人 000005108  
株式会社日立製作所  
東京都千代田区神田駿河台四丁目6番地  
(72) 発明者 小山 昌宏  
茨城県土浦市神立町502番地 株式会社日立製作所機械研究所内  
(72) 発明者 細田 祐司  
茨城県土浦市神立町502番地 株式会社日立製作所機械研究所内  
(74) 代理人 100093872  
弁理士 高崎 芳紘

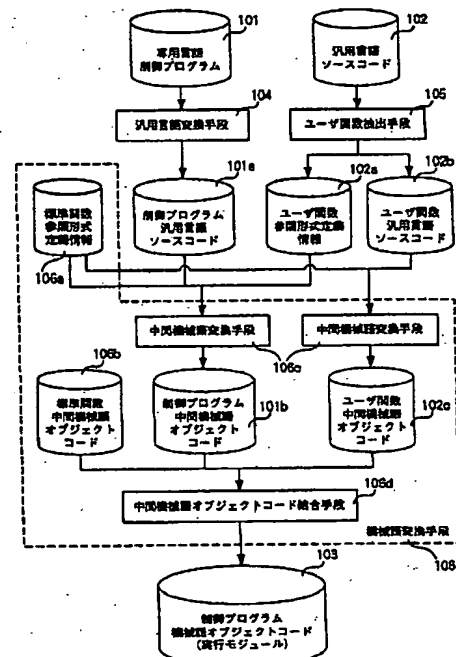
最終頁に続く

(54) 【発明の名称】 制御プログラム作成方法とその装置

(57) 【要約】

【課題】 ユーザ関数を引用するための余計な手間と入力ミスを削減できる制御プログラムの作成方法を提供する。

【解決手段】 ユーザ関数抽出手段105は、ユーザが汎用言語で記述したソースコード102から、制御プログラムの中で利用するユーザ関数の参照形式定義情報102aとユーザ関数汎用言語ソースコード102bを生成する。汎用言語変換手段104は、ユーザ関数の記述が混在した制御プログラム101を制御プログラム汎用言語ソースコード101aに変換し、機械語変換手段106は、定義情報102aとソースコード102bを用いて、ソースコード101aを機械語に変換する。



## 【特許請求の範囲】

【請求項 1】 問題向きの専用言語により記述された制御プログラムを制御装置が実行可能な機械語に変換するための制御プログラム作成装置において、

ユーザが汎用言語により記述したソースコード群から、制御プログラムの中で利用するユーザ関数の参照形式定義情報と、前記ユーザ関数に関連する処理のみを抽出したユーザ関数汎用言語ソースコードとを生成するユーザ関数抽出手段と、

専用言語の記述の中に前記ユーザ関数の記述を混在させた制御プログラムから制御プログラムの汎用言語ソースコードを生成する汎用言語変換手段と、

前記ユーザ関数抽出手段により生成された参照形式定義情報及びユーザ関数汎用言語ソースコードを用いて、前記汎用言語変換手段により生成された制御プログラムの汎用言語ソースコードを前記機械語に変換する機械語変換手段と、

を備えたことを特徴とする制御プログラム作成装置。

【請求項 2】 前記ユーザ関数抽出手段は、前記汎用言語により記述したソースコード群に含まれる関数の宣言文とこれらの関数対応のチェックボックスを持つチェックリストを生成、表示するチェックリスト表示手段を備え、チェックボックスにより選択されたユーザ関数に対して前記参照形式定義情報と前記ユーザ関数汎用言語ソースコードとを自動的に生成することを特徴とする請求項 1 記載の制御プログラムの作成装置。

【請求項 3】 前記専用言語は、ユーザ関数を利用して汎用言語により記述されたブロックを指示するための指示コマンドを有しており、前記汎用言語変換手段は、前記制御プログラムの、前記指示コマンドで指定されていない専用言語のコマンドを汎用言語の標準関数に変換し、前記指示コマンドで指示されたブロックの内容をそのままコピーすることにより制御プログラムの汎用言語ソースコードを生成することを特徴とする請求項 1 記載の制御プログラム作成装置。

【請求項 4】 前記制御プログラムは、シーケンシャル・ファンクション・チャートの図形形式の記述とテキスト形式の記述を組み合わせで作成したプログラムを、シーケンスの各ステップの状態遷移に対応した IF～THEN 形式のルールによる記述に変換して保存されたプログラムであり、前記汎用言語変換手段は、前記ルールの各々をステップが活性化されたときに実行される関数に変換して制御プログラムの汎用言語ソースコードを生成することを特徴とする請求項 1 記載の制御プログラム作成装置。

【請求項 5】 問題向きの専用言語により記述された制御プログラムを制御装置が実行可能な機械語に変換するための制御プログラム作成方法において、ユーザ関数を利用して汎用言語により記述されたブロックを、そのブロックを指示するための指示コマンドを用

いて制御プログラム中に記述し、

前記ユーザ関数の各々のソースコードから当該ユーザ関数の参照定義情報と当該ユーザ関数の処理を記述するユーザ関数汎用言語ソースコードとを生成し、

前記制御プログラムの、前記指示コマンドで指示されていない専用言語の記述を制御プログラム汎用言語ソースコードに変換した後、この制御プログラム汎用言語ソースコードを前記参照定義情報及び前記ユーザ関数汎用言語ソースコードを用いて機械語に変換することを特徴とする制御プログラム作成方法。

## 【発明の詳細な説明】

## 【0001】

【発明の属する技術分野】本発明は、問題向きの専用言語により記述した制御プログラムを翻訳して制御装置が実行可能な機械語を生成する制御プログラム作成方法とその装置に関する。

## 【0002】

【従来の技術】ロボット、プラント等の制御装置では、問題向きの専用言語（例えば、ロボット言語やラダー図など）により制御プログラムを記述し、この制御プログラムをコンパイラにより機械語に翻訳したのちに実行するか、あるいはこの制御プログラムをインタプリタにより逐次解釈及び実行する。

【0003】このような制御装置では、ユーザが容易に制御プログラムを作成できるように、各種制御機能を実現する標準関数ライブラリが用意されており、これを専用言語の命令として利用できる。また、この標準関数に含まれない制御機能をユーザが必要とする場合には、予め用意された専用言語の命令を組み合わせることでその機能を実現するか、あるいは何らかの方法で標準関数ライブラリにユーザの仕様に基づいた新しい関数（ユーザ関数）を追加して、これを専用言語の新しい命令として利用できるようにコンパイラあるいはインタプリタを改造する必要がある。

【0004】しかし、前者の専用言語の命令を組み合わせた記述による場合には、その記述能力に限界があり、ユーザの必要とする制御機能を記述できない可能性がある。あるいは、仮に記述できたとしても、記述が複雑になってしまい、プログラムとして読解性が低下する可能性がある。また、インタプリタによる実行の場合、このように専用言語により複雑な演算処理を記述すると、プログラムの解釈処理時間が増大して、制御システムの応答性が低下する可能性がある。一方、ユーザ関数をライブラリへ追加するためには、一般的にはメーカにソフトウェアの改造を依頼することになり、このために多額の費用が発生する可能性がある。また、ユーザが必要とする制御機能をユーザ自身が汎用言語によるプログラムとして所有している場合もあり、このような既存のソフトウェア資産を有効活用したいという要望もあったが、従来の制御装置ではこれを容易に実現できなかった。

【0005】このような問題点を解決する試みに関する公知例としては、特開平11-338521号公報に記載された制御装置が挙げられる。この公知例では、制御装置の基本ソフトウェアにユーザ関数登録機能を備える発明が述べられている。このユーザ関数登録機能では、ユーザがユーザ関数の名称と引数等を定義する文法、ユーザ関数の演算内容を汎用言語により記述したもの、さらにユーザ関数の実行モジュールのメモリ上でのエントリアドレスを入力することにより、専用言語の標準関数とは別のユーザ関数を登録できるようになっている。この公知技術によれば、制御プログラムの作成時にはユーザ関数をそのまま記述できるので、プログラムの解読性が低下することではなく、また制御プログラムの実行時にはユーザ関数の実行コードを呼び出すので、その解釈処理時間が大幅に増大することはないと考えられる。また、制御装置の改造を第三者に依頼する必要はなく、ユーザ自身が所望のユーザ関数を新たに登録することができる。

#### 【0006】

【発明が解決しようとする課題】上記した特開平11-338521号公報に記載されたユーザ関数登録機能では、ユーザ自身がユーザ関数の名称と引数等を定義する文法とユーザ関数の実行モジュールのエントリアドレス等を所定の書式にしたがって入力する必要があるため、ユーザに余計な手間をかけてしまうだけでなく、この段階で入力ミスが発生する可能性もあり、プログラム作成上の効率の面でさらなる改善が望まれる。また、ユーザが既に汎用言語によるソフトウェア資産を持っており、これを部分的に利用したいと考えている場合、上記ユーザ関数登録機能では、そのソースコードを再編集する必要がある可能性がある。さらに、上記公知例では、制御プログラムをインタプリタ方式により実行することを前提としているため、プログラム実行効率の面でも改善の余地はある。

【0007】そこで、本発明では、ユーザが汎用言語で作成した関数を、簡単な操作で制御プログラムに組み込んで利用でき、かつその組み込みにより実行速度が特に低下しない制御プログラム作成方法とその装置を提供することを目的とする。

#### 【0008】

【課題を解決するための手段】本発明は、問題向きの専用言語により記述された制御プログラムを制御装置が実行可能な機械語に変換するための制御プログラム作成装置において、ユーザが汎用言語により記述したソースコード群から、制御プログラムの中で利用するユーザ関数の参照形式定義情報と、前記ユーザ関数に関連する処理のみを抽出したユーザ関数汎用言語ソースコードとを生成するユーザ関数抽出手段と、専用言語の記述の中に前記ユーザ関数の記述を混在させた制御プログラムから制御プログラムの汎用言語ソースコードを生成する汎用言

語変換手段と、前記ユーザ関数抽出手段により生成された参照形式定義情報及びユーザ関数汎用言語ソースコードを用いて、前記汎用言語変換手段により生成された制御プログラムの汎用言語ソースコードを前記機械語に変換する機械語変換手段と、を備えたことを特徴とする制御プログラム作成装置を開示する。

【0009】更に本発明は、前記ユーザ関数抽出手段は、前記汎用言語により記述したソースコード群に含まれる関数の宣言文とこれらの関数対応のチェックボックスを持つチェックリストを生成、表示するチェックリスト表示手段を備え、チェックボックスにより選択されたユーザ関数に対して前記参照形式定義情報と前記ユーザ関数汎用言語ソースコードとを自動的に生成することを特徴とする制御プログラムの作成装置を開示する。

【0010】更に本発明は、前記専用言語は、ユーザ関数を利用して汎用言語により記述されたブロックを指示するための指示コマンドを有しており、前記汎用言語変換手段は、前記制御プログラムの、前記指示コマンドで指定されていない専用言語のコマンドを汎用言語の標準関数に変換し、前記指示コマンドで指示されたブロックの内容をそのままコピーすることにより制御プログラムの汎用言語ソースコードを生成することを特徴とする制御プログラム作成装置を開示する。

【0011】更に本発明は、前記プログラムは、シーケンシャル・ファンクション・チャートの図形形式の記述とテキスト形式の記述を組み合わせで作成プログラムを、シーケンスの各ステップの状態遷移に対応したIF～THEN形式のルールによる記述に変換して保存されたプログラムであり、前記汎用言語変換手段は、前記ルールの各々をステップが活性化されたときに実行される関数に変換して制御プログラムの汎用言語ソースコードを生成することを特徴とする制御プログラム作成装置を開示する。

【0012】更に本発明は、問題向きの専用言語により記述された制御プログラムを制御装置が実行可能な機械語に変換するための制御プログラム作成方法において、ユーザ関数を利用して汎用言語により記述されたブロックを、そのブロックを指示するための指示コマンドを用いて制御プログラム中に記述し、前記ユーザ関数の各々のソースコードから当該ユーザ関数の参照定義情報と当該ユーザ関数の処理を記述するユーザ関数汎用言語ソースコードとを生成し、前記制御プログラムの、前記指示コマンドで指示されていない専用言語の記述を制御プログラム汎用言語ソースコードに変換した後、この制御プログラム汎用言語ソースコードを前記参照定義情報及び前記ユーザ関数汎用言語ソースコードを用いて機械語に変換することを特徴とする制御プログラム作成方法を開示する。

#### 【0013】

【発明の実施の形態】以下、本発明の実施の形態につい

て、添付図面を参照して説明する。図1は、本発明になる制御プログラム作成装置の処理手順例を示す図である。この制御プログラム作成装置は、プログラマブルロジックコントローラ用のシーケンス制御言語、モーションコントローラ用のモーション制御言語、ロボットコントローラ用のロボット言語などの専用言語に用意された標準関数ライブラリを用いて記述された専用言語プログラム101中に、ユーザの作成した汎用言語ソースコード102中に含まれるユーザ関数を組み込み、それらを最終的に機械語に翻訳して制御プログラム機械語オブジェクトコード103を生成するものである。

【0014】このために、図1の制御プログラム作成装置には、汎用言語ソースコード102から、制御プログラムの中でユーザ関数を利用するためのユーザ関数参照形式定義情報102aと、ユーザ関数に関連する処理のみを抽出したユーザ関数汎用言語ソースコード102bとを自動的に生成するユーザ関数抽出手段105を設ける。ここで抽出されたユーザ関数は、専用言語制御プログラム101の中で記述することが可能となる。このように専用言語の記述の中にユーザ関数の記述を混在させた専用言語制御プログラム101は、汎用言語変換手段104によって制御プログラム汎用言語ソースコード101aに変換される。機械語変換手段106は、以上のように生成された制御プログラム汎用言語ソースコード101a、ユーザ関数参照形式定義情報102a、及びユーザ関数汎用言語ソースコード102bから、最終的に制御装置内で実行可能な形式の制御プログラム機械語オブジェクトコード103を生成する。以下では、汎用言語としては一般に広く普及しているC言語を用いるものとする。

【0015】機械語変換手段106では、制御プログラム汎用言語ソースコード101a、ユーザ関数汎用言語ソースコード102bは、中間機械語変換手段106cによって、それぞれ制御プログラム中間機械語オブジェクトコード101b、ユーザ関数中間機械語オブジェクトコード102cに変換される。ここで、制御プログラム汎用言語ソースコード101aにはユーザ関数が記述されており、このユーザ関数の実体は別のソースコードであるユーザ関数汎用言語ソースコード102bに記述されている。このようなユーザ関数の外部参照を可能とするためにユーザ関数参照形式定義情報102aが必要となり、制御プログラム汎用言語ソースコード101aを中間機械語に変換する際に用いられる。これと同様に、専用言語及び汎用言語に含まれる各種標準関数の実体は、汎用言語処理システムに備えられている標準関数ライブラリに含まれており、これらの標準関数を外部参照するために標準関数参照形式定義情報106aが必要であり、制御プログラム汎用言語ソースコード101a、ユーザ関数汎用言語ソースコード102bを中間機械語に変換する際に用いられる。

【0016】中間機械語オブジェクトコード結合手段106dは、制御プログラム中間機械語オブジェクトコード101b、ユーザ関数中間機械語オブジェクトコード102c、さらに標準関数ライブラリを中間機械語に変換した標準関数中間機械語オブジェクトコード106bから、標準関数及びユーザ関数の外部参照を解決すると同時にそれぞれの中間機械語オブジェクトコードを結合して、最終的な実行モジュールである制御プログラム機械語オブジェクトコード103を生成する。

【0017】前記のように汎用言語としてC言語を用いるものとする、制御プログラム汎用言語ソースコード101a、ユーザ関数汎用言語ソースコード102bは、それぞれC言語のソースコードであり、標準関数参照形式定義情報106a、ユーザ関数参照形式定義情報102aは、それぞれ標準関数、ユーザ関数のC言語のextern宣言を記述したヘッダファイル（インクルードファイル）に相当する。なお、これらのヘッダファイルは制御プログラム汎用言語ソースコード101aの冒頭部分において、プリプロセッサ命令#includeによって取り込まれる。また、中間機械語変換手段106c、及び中間機械語オブジェクトコード結合手段106dは、それぞれC（クロス）コンパイラ、及びリンカに相当する。

【0018】以上の図1に示したプログラム作成装置において、本発明の特徴とするのは、ユーザ関数抽出手段105及び汎用言語変換手段104であり、以下、これらを中心とした動作を具体例を用いて説明する。

【0019】図2は、制御プログラムの編集画面の一例を示す図である。ここで、制御プログラムの処理の流れ（シーケンス）はSFC（シーケンシャル・ファンクション・チャート）により記述し、シーケンスの各ステップの処理はテキスト形式の専用言語により記述する。シーケンス編集ウィンドウ201では、制御対象となる各機器のシーケンスをSFCの図形形式により記述する。コマンド編集ウィンドウ202では、シーケンスの各ステップの動作を専用言語のコマンドとして記述する。終了条件編集ウィンドウ203では、各ステップが終了するための条件を条件式として記述する。シーケンス編集ウィンドウ201で記述されるSFCのステップS100、S101…の各々は各機器の単位動作を示し、その動作内容はコマンド列としてコマンド編集ウィンドウ202において記述される。

【0020】図2右側のコマンド編集ウィンドウ202には、ステップS103のコマンド列が記述されており、これとポジションの目標位置を計算するためのパラメータを取得し、ユーザ関数を用いて目標位置を計算し、さらに、この目標位置へポジションを移動させ、ワークを把持するためのチェックを開いて、チェック開動作を監視するためのオンディレイタイマを起動するステップである。また、トランジション201b、201c

は、ステップ間の状態遷移を示し、その遷移条件はこれらのトランジションの入力ステップS103の終了条件として、終了条件編集ウィンドウ203に記述される。ステップS103の終了条件は2種類あり、制限時間内にチェックが完全に開いた場合（終了条件T1）、制限時間内にチェックが完全に開かない場合（終了条件T2）を定義する条件式が記述されている。終了条件T1、T2はステップS103から次のステップに遷移するための条件として、その出力トランジション201b、201cの右横に付記される。

【0021】コマンド編集ウィンドウ202では、制御装置側で予め用意された専用言語のコマンドを記述するだけでなく、ユーザがC言語などの汎用言語により作成したユーザ関数を記述することができる。図1の構成では、このようにコマンド列の中にC言語によるユーザ関数を記述した部分（以下、ユーザ関数ブロック）を明示するためにコマンド#C、#END\_Cが用意されており、これらで囲まれた部分がユーザ関数ブロックを表している。図2のコマンド編集ウィンドウ202に示したコマンド列の中にある#Cと#END\_Cで囲まれた部分、すなわちユーザ関数ブロック202aには、C言語で別途定義されたユーザ関数calc\_pos\_x()、calc\_pos\_y()によりポジションの目標位置を計算し、これらを専用言語の変数VR4、VR5（それぞれC言語表記では、VR[4]、VR[5]）に代入する処理が記述されている。変数VR4、VR5は、ユーザ関数ブロック202aに続く専用言語のコマンド(MOVE AXIS1 VR4 AXIS2 VR5 F 2000.)のパラメータとして参照されている。なお、ここに示すように、ユーザ関数ブロック202aの中の表記は、C言語の文法に従った表記となる。また、専用言語で用意される番号付きの各種変数(VRnなど)はユーザ関数ブロック202aの中でも利用可能であり、これらはC言語の配列(VR[n]など)として表記される。

【0022】このようなユーザ関数ブロック202aを示すコマンドを設けたことにより、ユーザが慣れ親しんだC言語の文法に従って容易にユーザ関数を記述することが可能となる。また、このようにユーザ関数ブロック202aを明示するコマンドを利用すれば、後述のように、専用言語制御プログラム101から制御プログラム汎用言語ソースコード101aに変換する汎用言語変換手段104の処理を簡単にすることができる。

【0023】なお、図2のようなSFCでは、並行して実行される各シーケンスの間の同期を表現するためのステップが用意されており、これを同期ステップと呼ぶ。例えば、シーケンス編集ウィンドウ201に示した同期ステップREQ1は、ステップS103の入力トランジション(201f)に対する入力となっており、これは入力トランジション(201f)において他のシーケン

スが出力する同期信号を待つことを意味する。すなわち、ステップS103への遷移は、直前のステップS102が終了条件T1で終了し、さらに同期ステップREQ1が活性化されることにより発生する。

【0024】図4は、ユーザ関数抽出手段105の概略処理を示すフローチャート、図3はその処理例を示す説明図である。通常、ユーザに関連するユーザ関数を記述したソースコード群を1つのファイルにまとめているから、この中のいくつかのユーザ関数を利用するときは、まずユーザが該当するファイルを指定して記述されたソースコードをオープンする(処理401)。図3のC言語のソースコード301はその記述例で、float型の変数x、y、cからfloat型の値を算出する関数calc\_vel\_x、calc\_vel\_y、calc\_pos\_x、calc\_pos\_y…が記述されている。但し、各関数の処理を定義する{…}の部分は、通常複数の行に分けて記述されるが、図3ではこれを

{…}の一行で表している。次に、表示されたソースコードから、その中のユーザ関数を調べて、ユーザ関数のチェックリスト302を作成し、これをユーザに対して提示する(処理402)。このチェックリスト302は、関数宣言文と各関数に設けたチェックボックスより成っている。ユーザはこのチェックリストを参照し、制御プログラムの中で使用するユーザ関数をチェックボックスをクリックして選択する(処理403)。

【0025】ユーザ関数の選択が終了すれば、選択されたユーザ関数の中で直接的または間接的に呼び出されるすべての関数を調べ(処理404)、選択されたユーザ関数とそれに関連する関数のソースコードを抜き出した新たなユーザ関数ソースコード304を生成する(処理405)。さらに、ユーザ関数のチェックリスト302のユーザ関数宣言文から、選択されたユーザ関数の外部参照宣言ヘッダファイル303を生成する(処理406)。ここで生成されたユーザ関数生成コード304及び外部参照ヘッダファイル303は、それぞれ図1のユーザ関数汎用言語ソースコード102b及びユーザ関数参照形式定義情報102aに相当する。また外部参照ヘッダファイル303は、選択されたユーザ関数の関数宣言文に外部参照可能であることを示す“extern”を付加したリストである。

【0026】以上に述べたユーザ関数抽出手段105を利用すれば、ユーザはC言語ソースコード301から生成されるユーザ関数のチェックリスト302にチェックするという簡単な操作を行うだけで選択したユーザ関数を専用言語プログラム中で利用できるようになるので、ユーザ関数を引用するためのユーザの手間と入力ミスを極力省くことができる。また、ユーザが既に汎用言語によるソフトウェア資産を所有している場合、これを容易かつ有効に活用することが可能となる。

【0027】図6は、汎用言語変換手段104の概略処

理を示すフローチャート、図5はその処理例を示す図で、図2に示した専用言語制御プログラム101の一部を制御プログラム汎用言語ソースコード101aに変換する場合を示している。図2で説明したように、制御プログラムをSFCで作成するときには、制御プログラムのシーケンスを図2のウィンドウ201のようなSFCの図形形式で記述し、シーケンスの各ステップの動作と終了条件を図2のウィンドウ202のようにテキスト形式の言語により記述する。このような図形形式とテキスト形式を組み合わせる記述された専用言語制御プログラム101は、図5に示すようなテキスト形式の専用言語制御プログラム510に自動的に変換され、ファイルとして保存されている。

【0028】この専用言語制御プログラム510はIF～THEN形式のルールによって記述されており、各ルールの条件部にはシーケンスのあるステップから次のステップへの状態遷移を定義する条件、実行部には遷移した次のステップで実行されるコマンド列が記述される。例えば、ルール511の条件部511aには、ステップS102が正常終了(Complete Normally)し、さらに同期ステップREQ1が活性化される(ON)という条件式が記述されている。ルール511の実行部の最初のブロック511bは、ステップの遷移に関するコマンド列であり、ステップS102と同期ステップREQ1を不活性状態にし(TERM S102、TERM REQ1)、ステップS103を実行中状態にする(EXEC S103)。次に続くブロック511c、511d、511eは、図2のコマンド編集ウィンドウ202に示したステップS103のコマンド列であり、ブロック511c、511eは専用言語によるコマンド列、ブロック511dはC言語によるユーザ関数ブロック202aである。最後のEND\_IF文511fは、ルール511の終わりを示す。

【0029】汎用言語変換手段104は、まず、ユーザにより指定された専用言語制御プログラム101のファイルをオープンし(処理601)、上記のような形式で記述された専用言語制御プログラム510を一行ずつ取り込む(処理602)。次に取り込んだ行がIF文かを調べ(処理603)、IF文であればこの行から次に現れるEND\_IF文までのルールを1つの関数とすべく、IF文の第1条件式から関数宣言部(関数の型と名称)を生成し(処理604)、さらにこの第1条件式からif条件文を生成する(処理605)。ここで生成されるルール対応の関数は、各ステップが活性化されたときに実行される関数であり、関数名は活性化されるステップ番号に対応している。図5のルール511の場合には、図2のステップS102が活性化(ステップS102のすべてのコマンド発行が終わっている状態)していることが第1条件式であるので、処理604ではルール511対応の関数名はステップS102に対して“st

t102”とされ、そして処理605では、第1条件式から対応するif条件文“if(stt[102]==Complete Normally)”が生成される。

【0030】次の一行を取り込み、これがIF文でなかったときは(処理603でNO)、それが&(アンド)条件であれば(処理606でYES)、その条件式からif条件文の続きを生成し(処理607)、&条件でないときは処理608へ移行する。図5の例では、2行目はIF文の続きの&条件“& REQ1=ON”であるから、これに対応するif条件式“&& Req[1]==ON”が生成される。処理607は&条件がなくなるまで繰り返される。こうして、ルール511の条件部511aは、C言語のif文に変換され、ブロック521aが生成される。

【0031】取り込んだ行がIF文でも&条件でもないときは(処理603、606でNO)、その行がTHEN文かを判定し(処理608)、THEN文でないときは注釈行等の処理に関係ない文として無視して処理602へ戻る。処理608の判定結果がTHEN文であれば、if文の実行部の生成に移る。ここで、さらに1行ずつ取り込み(処理609)、これが#C文でなく(処理610でNO)、END\_IF文でもなければ(処理611でNO)、これは専用言語のコマンドであり、これをC言語の標準関数に変換する(処理612)。また、取り込んだ行が「#C」であれば(処理610でYES)、ここからユーザ関数ブロックが始まることになり、さらに1行ずつ取り込んで(処理614)、これが#END\_C文でなければ(処理615でNO)、取り込んだ文字列、すなわちC言語による表記をそのままコピーする(処理616)。この処理を#END\_C文に到達するまで(処理615でYES)繰り返す。このようにユーザ関数ブロックの終わりに到達すれば、再び1行ずつ取り込み(処理609)、これがEND\_IF文になるまで、以上のようなif文の実行部の生成処理を繰り返す(処理611でYES)。END\_IF文に到達すれば(処理611)、if文の終了部分と関数の終了部分を生成する(処理613)。以上のような変換の処理を最終行に到達するまで繰り返す(処理614)。

【0032】以上実行部生成処理(図6の処理608～614)を図5のルール511に対して実行すると、関数521のif文の実行部に当たるブロック521b、521c、521d、521eが、それぞれルール511のブロック511b、511c、511d、511eから生成され、このうちブロック521b、521c、521eは専用言語のコマンド列を対応するC言語の標準関数に変換したもの、ブロック521dはC言語によるユーザ関数ブロック511dの内容をコピーしたものとなっている。また関数521の最終部分のブロック521fは、ルール511の最後のEND\_IF文に対応して処理613で生成される。図5の続くルール512

からも同様にして関数522が生成される。

【0033】以上のように、制御プログラムを汎用言語に変換することにより、汎用言語によるユーザ関数の引用が容易になり、さらにこれらの汎用言語のソースコードをコンパイル、リンクすることで、制御装置において高速実行可能な機械語のオブジェクトコードを生成することが可能となる。

【0034】なお、本実施の形態では汎用言語としてC言語を用いた例を示したが、別の汎用言語（例えば、C++など）を用いて同様の制御プログラム作成装置を実現することも可能である。また、本実施の形態ではユーザ関数を記述するための汎用言語と専用言語を変換して得られる汎用言語を同じC言語とした例を示したが、これをそれぞれ異なった汎用言語（例えば、前者をアセンブリ言語、後者をC言語など）とすることも可能である。この場合、ユーザ関数汎用言語ソースコードと制御プログラム汎用言語ソースコードを、それぞれ別の中間機械語変換手段（コンパイラ）により中間機械語オブジェクトコードに変換し、それらを中間機械語オブジェクトコード結合手段により結合することで、同様の制御プログラムの作成方法を実現することが可能である。

#### 【0035】

【発明の効果】本発明によれば、ユーザが汎用言語ソースコードに含まれる関数の中から必要なユーザ関数を選択するだけで、専用言語の制御プログラムの中にユーザ関数の記述を混在させるための情報が自動的に生成されるので、ユーザ関数を引用するための余計な手間と入力ミスを削減し、プログラム作成効率を向上することが可能となる。さらに、本発明によれば、制御プログラムから制御装置のマイクロプロセッサが直接実行可能なオブジェクトコードが生成されるので、プログラム実行効率を向上することも可能となる。

#### 【図面の簡単な説明】

【図1】本発明になる制御プログラム作成装置の処理手順例を示す図である。

【図2】制御プログラムの編集画面の一例を示す図である。

【図3】ユーザ関数抽出手段による処理の一例を示す。

【図4】ユーザ関数抽出手段の概略処理を示すフローチャートである。

【図5】汎用言語変換手段により、図2に示した専用言語制御プログラムの一部を制御プログラム汎用言語ソースコードに変換した例を示す図である。

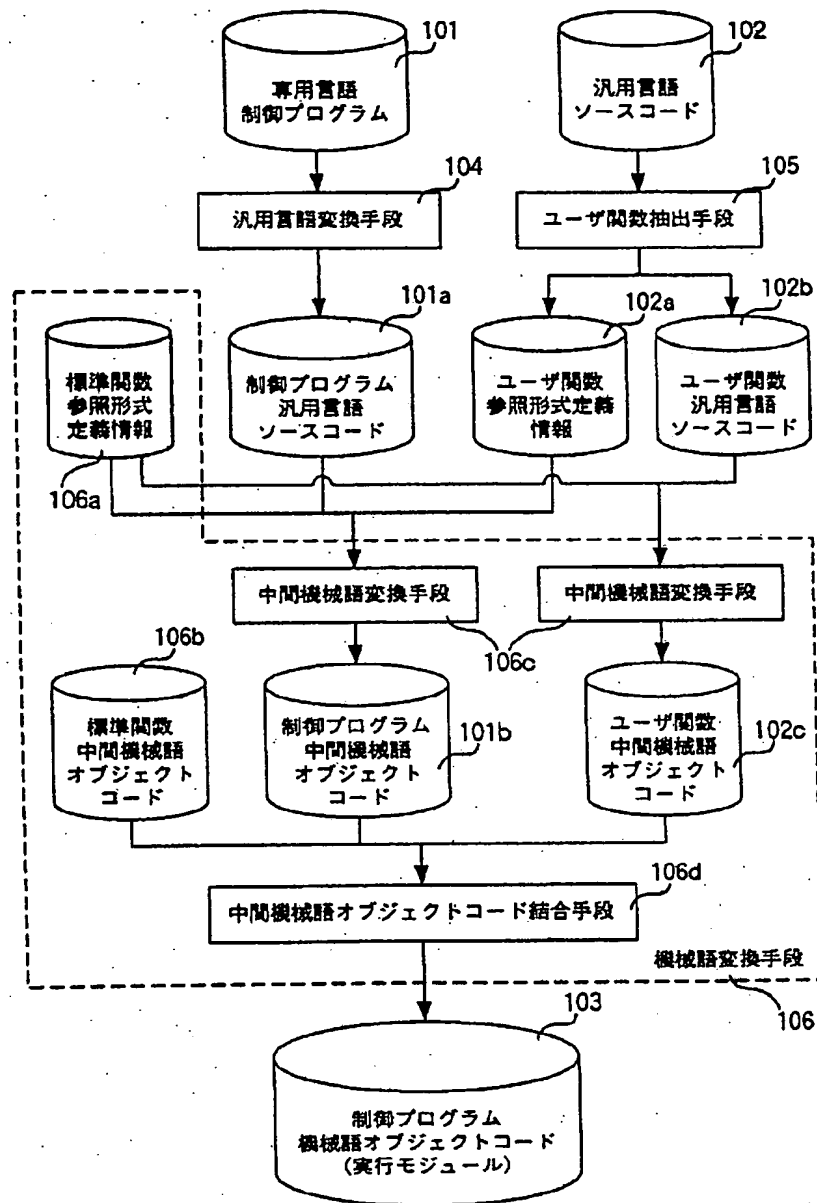
【図6】汎用言語変換手段の概略処理を示すフローチャートである。

#### 【符号の説明】

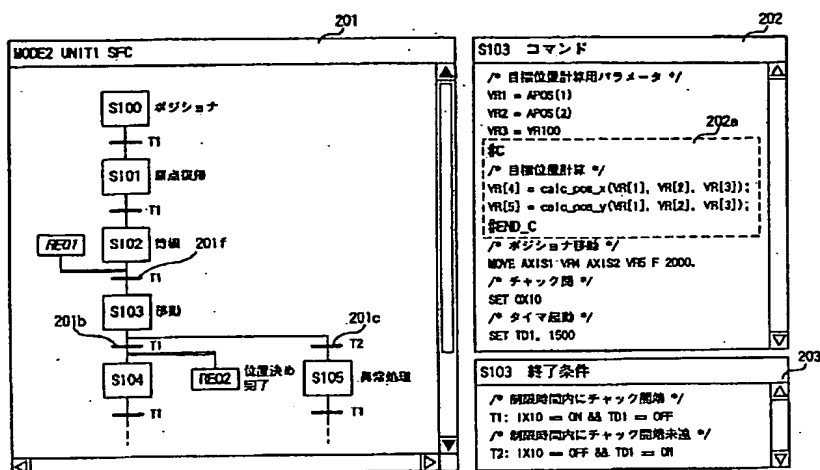
- 101 専用言語制御プログラム
- 101a 制御プログラム汎用言語ソースコード
- 101b 制御プログラム中間機械語オブジェクトコード
- 102 汎用言語ソースコード
- 102a ユーザ関数参照形式定義情報
- 102b ユーザ関数汎用言語ソースコード
- 102c ユーザ関数中間機械語オブジェクトコード
- 103 制御プログラム機械語オブジェクトコード
- 104 汎用言語変換手段
- 105 ユーザ関数抽出手段
- 20 106 機械語変換手段
- 106a 標準関数参照形式定義情報
- 106b 標準関数中間機械語オブジェクトコード
- 106c 中間機械語変換手段
- 106d 中間機械語オブジェクトコード結合手段
- 201 シーケンス編集ウィンドウ
- 202 コマンド編集ウィンドウ
- 202a ユーザ関数ブロック
- 203 終了条件編集ウィンドウ
- 301 C言語ソースコード
- 30 302 ユーザ関数のチェックリスト
- 302a チェックボックス
- 302b ユーザ関数の宣言文
- 303 ユーザ関数の外部参照宣言ヘッダファイル
- 304 ユーザ関数ソースコード
- 510 専用言語制御プログラム
- 511 ルール
- 520 制御プログラム汎用言語ソースコード
- 521 関数



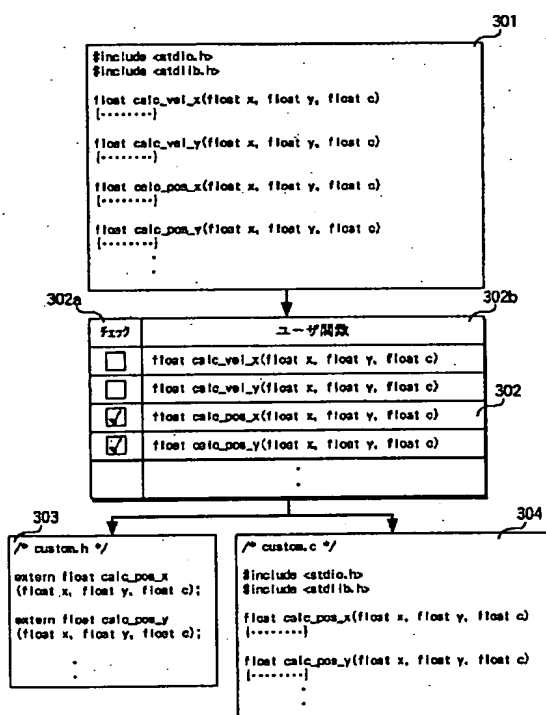
【図1】



【図2】

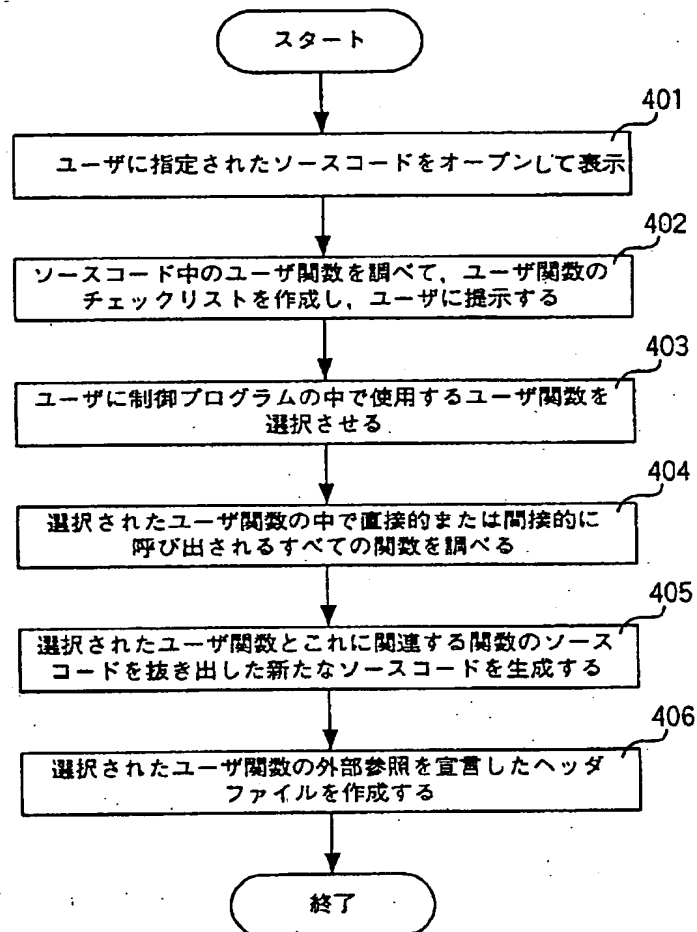


【図3】

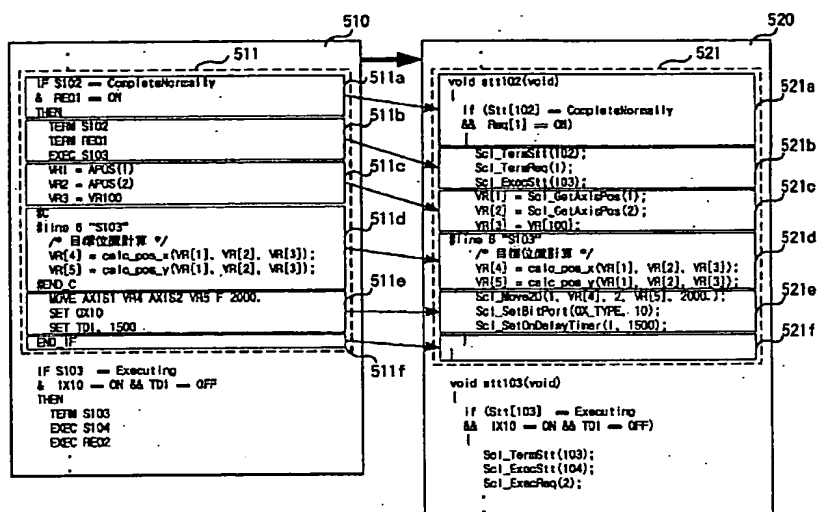


【図4】

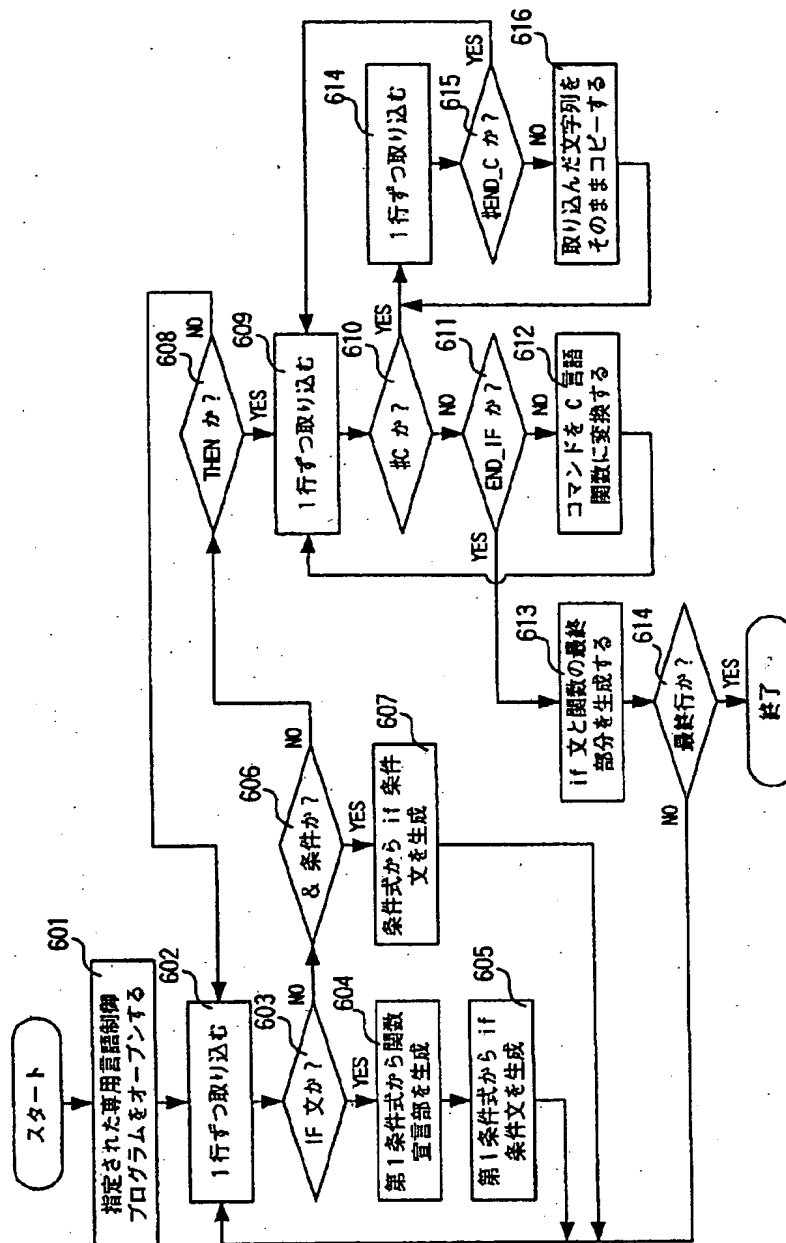
## ユーザ関数抽出手段の処理



【図 5】



【図6】



フロントページの続き

(72)発明者 松下 鶴正  
東京都千代田区神田駿河台四丁目6番地  
株式会社日立製作所内

(72)発明者 下津 忠夫  
東京都千代田区神田駿河台四丁目6番地  
株式会社日立製作所内

Fターム(参考) 5B081 AA10 CC01  
5H220 BB12 CC05 CX02 DD04 DD07  
JJ12 JJ24 JJ53